SSCM Exercise 4

Nikolaus Czernin - 11721138

```r
library("tidyverse")
```

```
## -- Attaching core tidyverse packages ------------------------ tidyverse 2.0.0 --
## v dplyr     1.1.2     v readr     2.1.4
## v forcats   1.0.0     v stringr   1.5.0
## v ggplot2   3.4.2     v tibble    3.2.1
## v lubridate 1.9.2     v tidyr     1.3.0
## v purrr     1.0.1
## -- Conflicts ------------------------------------------ tidyverse_conflicts() --
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()
## i Use the conflicted package (<http://conflicted.r-lib.org/>) to force all conflicts to become error
```

```r
library(knitr)

# Custom print function
print_ <- function(...) print(paste(...))

set.seed(11721138)
```

```r
# Define our starting sample X
X <- "4.94 5.06 4.53 5.07 4.99 5.16 4.38 4.43 4.93 4.72 4.92 4.96" %>%
  strsplit(" ") %>% unlist() %>% as.numeric() %>% print()
```

```
##  [1] 4.94 5.06 4.53 5.07 4.99 5.16 4.38 4.43 4.93 4.72 4.92 4.96
```

```r
N <- length(X)
```

Being lazy, I used strsplit() to get the values from the given sample text.

```r
ecdf <- function(n) c(0, seq(1,n)/n)
```

Task 1

Number of bootstrap samples

If we had a sample $X = 1, 2, 3$, with N=3, we would generate a bootstrap sample by sampling from X with replacement. The first number in the new sample would have thus N=3 possible values. For each value of the first sample, each other value could have also 1 out of 3 values. The number of different samples is therefore $3 * 3 * 3 = N^N$, which gives us the formula for any size of N (skipping the formal proof here).
The number of possible bootstrap samples is therefore $N^N = 12^12$ =8.9161004×10^{12}.

Mean and median

```r
X.mean <- mean(X)
X.mean %>% print_("Mean of X:", .)
```

```
## [1] "Mean of X: 4.84083333333333"
```

```
X.median <- median(X)
X.median %>% print_("Median of X:", .)
```

```
## [1] "Median of X: 4.935"
```

Generate 2000 bootstrap samples

```
m <- 2000
bootstrap.samples <- lapply(1:m, function(x) sample(X, replace = TRUE))
```

Analyse their means

```
bootstrap.means <- sapply(bootstrap.samples, function(s) mean(s))
```

I use lapply to loop 2000 times and create a new sample with replacement every time and then save the means of the sample using sapply instead.

```
mean20 <- mean(bootstrap.means[1:20]) %>%
  print_("Mean of the first 20 sample means:", .)
```

```
## [1] "Mean of the first 20 sample means: 4.8155"
```

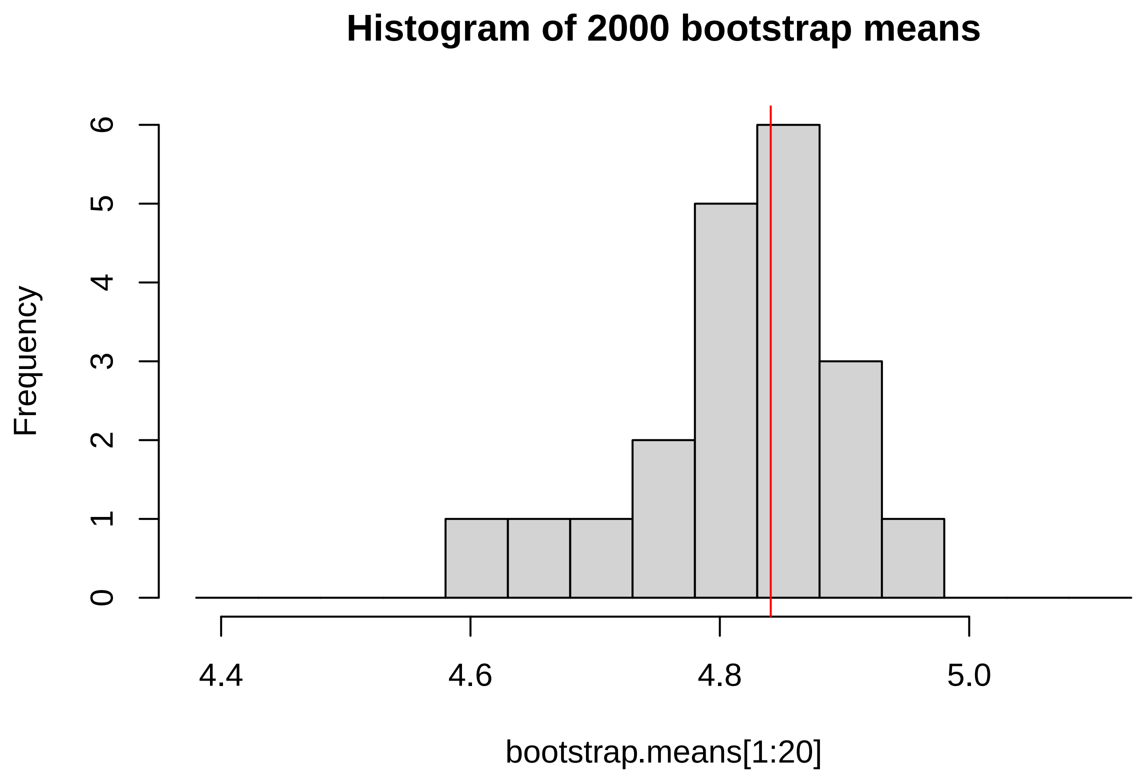
```
mean200 <- mean(bootstrap.means[1:200]) %>%
  print_("Mean of the first 200 sample means:", .)
```

```
## [1] "Mean of the first 200 sample means: 4.84145416666667"
```

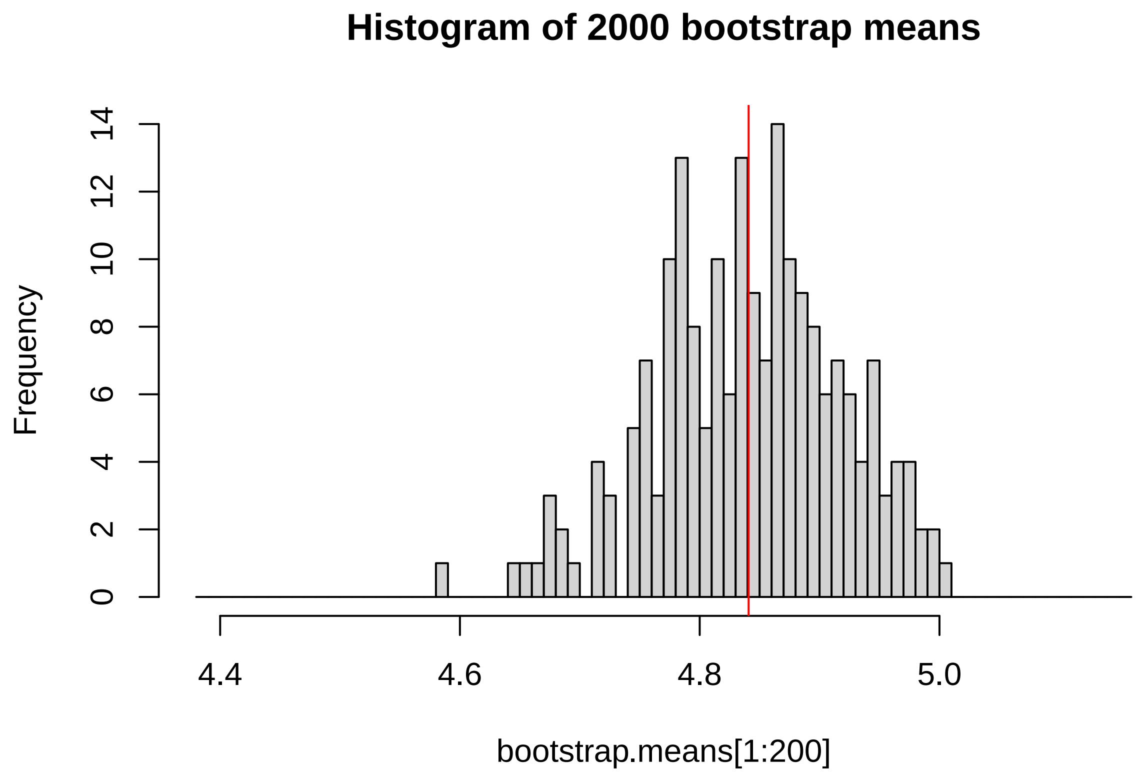
```
mean2000 <- mean(bootstrap.means[1:2000]) %>%
  print_("Mean of the first 2000 sample means:", .)
```

```
## [1] "Mean of the first 2000 sample means: 4.84045958333333"
```

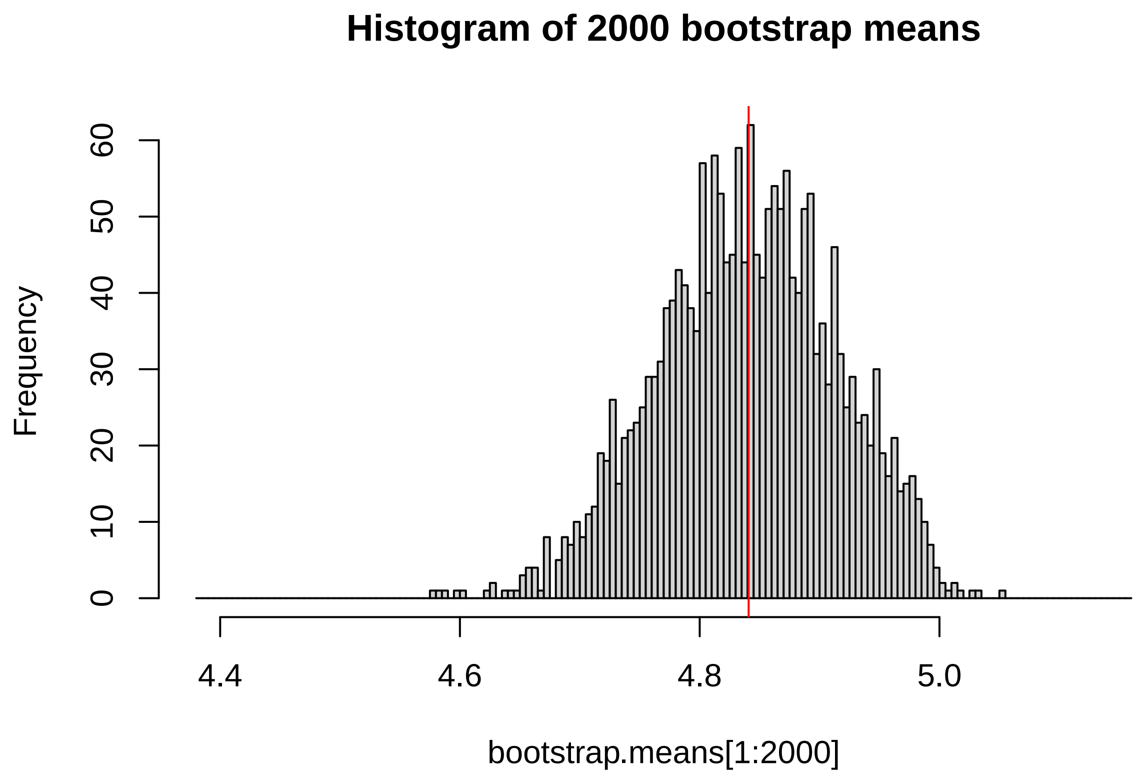
```
hist(bootstrap.means[1:20], breaks = seq(min(X), max(X), 0.05),
     main="Histogram of 2000 bootstrap means")
abline(v=X.mean, col="red")
```



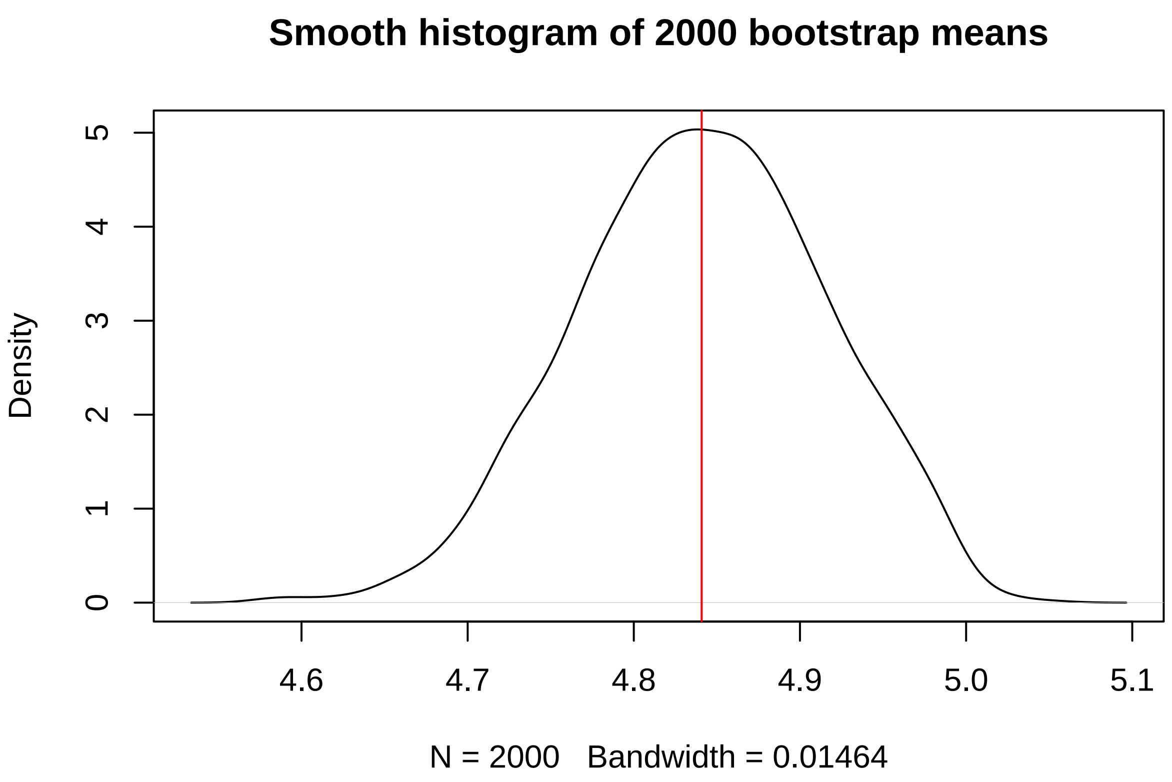
```r
hist(bootstrap.means[1:200], breaks = seq(min(X), max(X), 0.01),
     main="Histogram of 2000 bootstrap means")
abline(v=X.mean, col="red")
```



```
hist(bootstrap.means[1:2000], breaks = seq(min(X), max(X), 0.005),
     main="Histogram of 2000 bootstrap means")
abline(v=X.mean, col="red")
```



```
density(bootstrap.means) %>% plot(main="Smooth histogram of 2000 bootstrap means")
abline(v=X.mean, col="red")
```



At increasing sample size m we cna clearly observe the distribution of the means steadily getting the shape of a normal distirbution, outlining the central limit theorem.
I also plotted the curve using the density() of the means and plotting it as a line.

```
data.frame(
  k = c(20, 200, 2000, "True confidence interva"),
  Q025 = c(
    quantile(bootstrap.means[1:20], 0.025),
    quantile(bootstrap.means[1:200], 0.025),
    quantile(bootstrap.means[1:2000], 0.025),
    t.test(X)$conf.int[1]
    ),
  Q975 = c(
    quantile(bootstrap.means[1:20], 0.975),
    quantile(bootstrap.means[1:200], 0.975),
    quantile(bootstrap.means[1:2000], 0.975),
    t.test(X)$conf.int[2]
    )
)
```

```
##                         k     Q025     Q975
## 1                      20 4.626563 4.931729
## 2                     200 4.672500 4.980062
## 3                    2000 4.694979 4.978333
## 4 True confidence interva 4.674344 5.007323
```

By chance, the quantiles of the mean of the first 200 means appear to be an even close match to the “true” distributions, or rather a normal distribution with the same mean and standard deviation as the original sample, than the full 2000 bootstrap samples.

Analyse their medians

```
bootstrap.medians <- sapply(bootstrap.samples, function(s) median(s))
```

```
mean.median20 <- mean(bootstrap.medians[1:20]) %>%
  print_("Mean of the first 20 sample medians:", .)
```

```
## [1] "Mean of the first 20 sample medians: 4.88725"
```

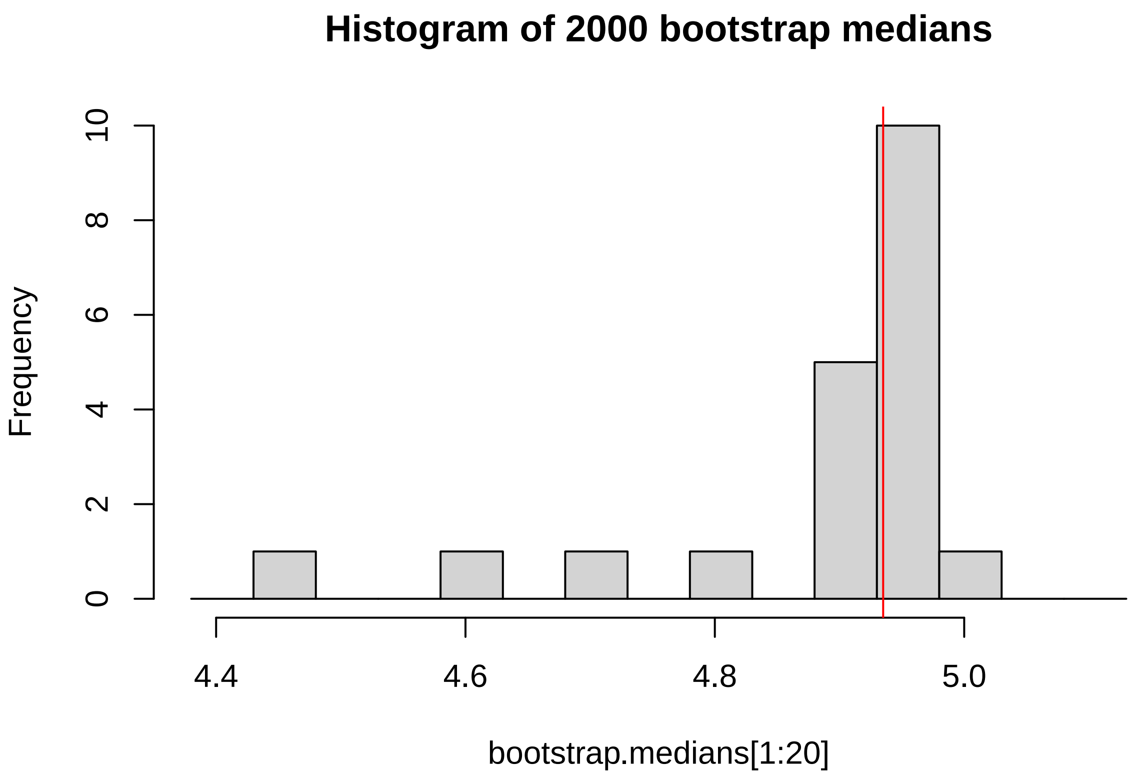
```
mean.median200 <- mean(bootstrap.medians[1:200]) %>%
  print_("Mean of the first 200 sample medians:", .)
```

```
## [1] "Mean of the first 200 sample medians: 4.91225"
```

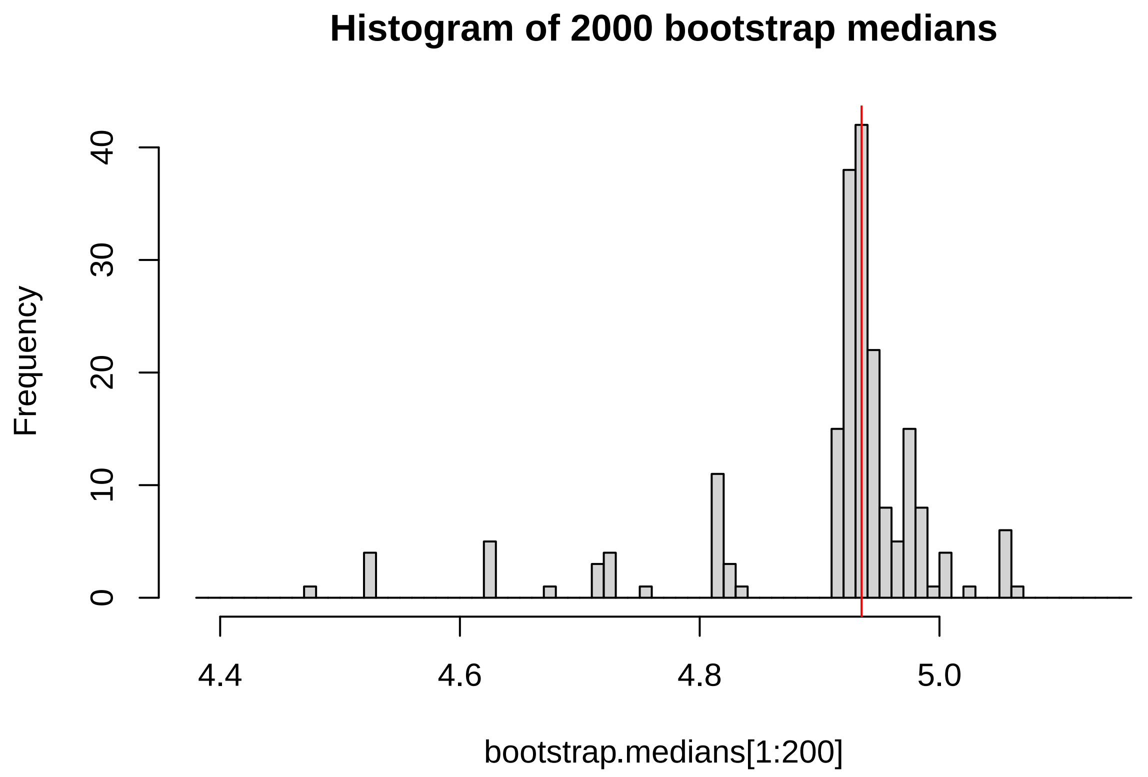
```
mean.median2000 <- mean(bootstrap.medians[1:2000]) %>%
  print_("Mean of the first 2000 sample medians:", .)
```

```
## [1] "Mean of the first 2000 sample medians: 4.911625"
```

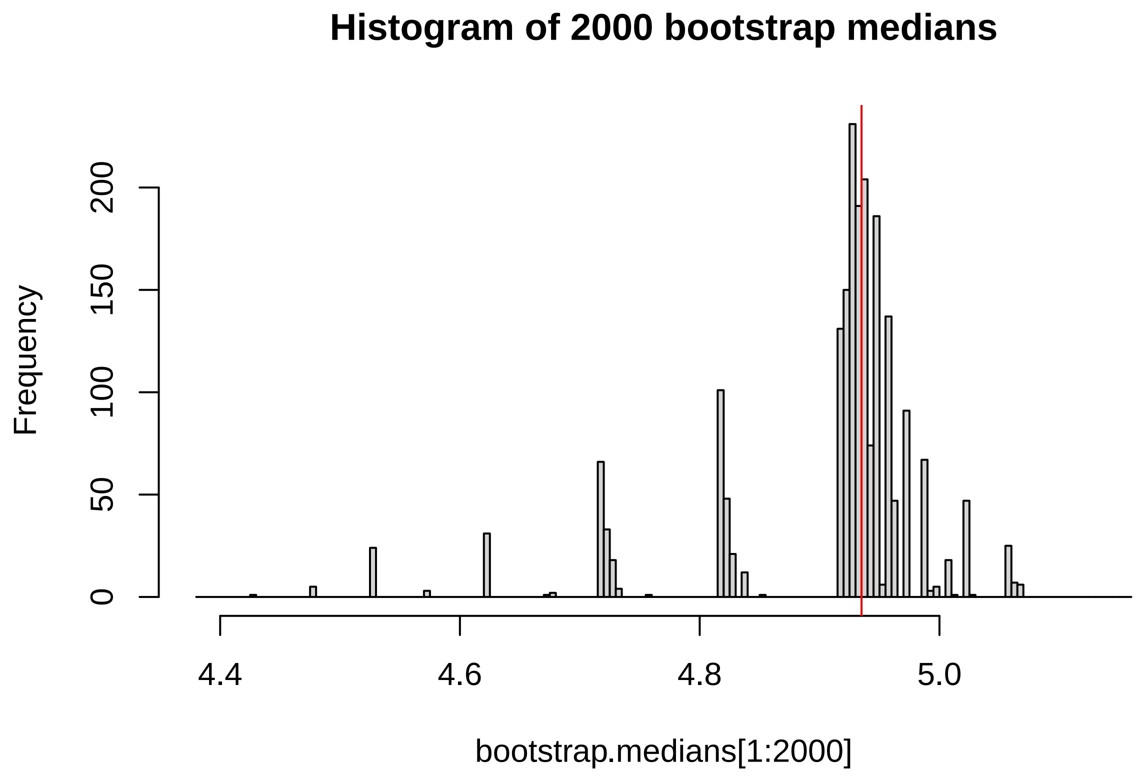
```
hist(bootstrap.medians[1:20], breaks = seq(min(X), max(X), 0.05),
     main="Histogram of 2000 bootstrap medians")
abline(v=X.median, col="red")
```



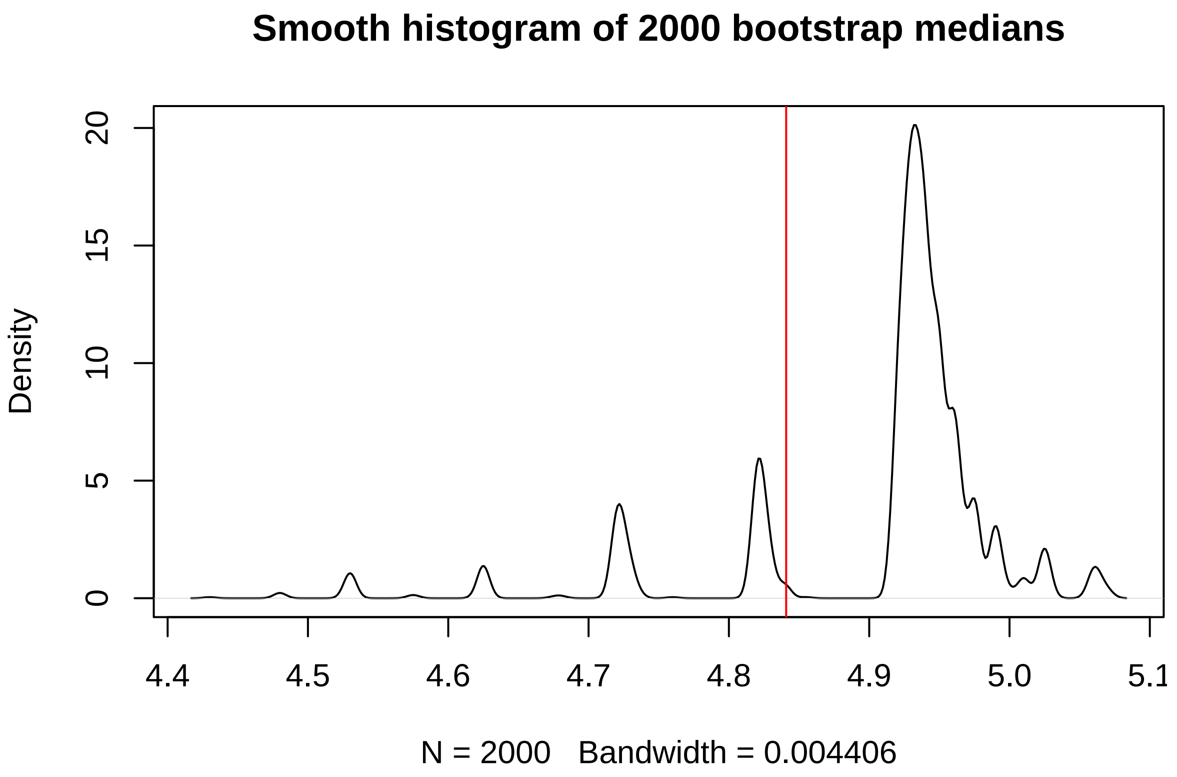
```
hist(bootstrap.medians[1:200], breaks = seq(min(X), max(X), 0.01),
     main="Histogram of 2000 bootstrap medians")
abline(v=X.median, col="red")
```



```
hist(bootstrap.medians[1:2000], breaks = seq(min(X), max(X), 0.005),
     main="Histogram of 2000 bootstrap medians")
abline(v=X.median, col="red")
```



```
density(bootstrap.medians) %>%
  plot(main="Smooth histogram of 2000 bootstrap medians")
abline(v=X.mean, col="red")
```



Regarding the medians, the Central Limit Theorem does not show at all, regarding the overall mean of medians. With more samples I feel like there might be a pattern of multiple little curves in the shape of normal distirbutions for each value in the original sample.
I also plotted the curve again using the density() of the medians and plotting it as a line.

```
data.frame(
  k = c(20, 200, 2000, "True confidence interval"),
  Q025 = c(
    quantile(bootstrap.medians[1:20], 0.025),
    quantile(bootstrap.medians[1:200], 0.025),
    quantile(bootstrap.medians[1:2000], 0.025),
    t.test(X)$conf.int[1]
    ),
  Q975 = c(
    quantile(bootstrap.medians[1:20], 0.975),
    quantile(bootstrap.medians[1:200], 0.975),
    quantile(bootstrap.medians[1:2000], 0.975),
    t.test(X)$conf.int[2]
    )
)
```

```
##                          k     Q025     Q975
## 1                       20 4.548875 4.982875
## 2                      200 4.622625 5.060000
## 3                     2000 4.625000 5.025000
## 4 True confidence interval 4.674344 5.007323
```

The mean medians' quantiles dont match the confidence interval of the original distribution as closely as the mean means did.

Task 2

Generate sample

```r
set.seed(1234)

x.clean <- rnorm(1960)
x.cont <- runif(40, min=4, max=5)
# total data
x <- c(x.clean,x.cont)

set.seed(11721138)
```

Estimate mean and median

```r
alpha = 0.05

x.median <- median(x)
x.mean <- mean(x)
x.mean.trimmed <- mean(x, trim = 0.05)

x.clean.median <- median(x)
x.clean.mean <- mean(x)
x.clean.mean.trimmed <- mean(x, trim = 0.05)
```

```r
m <- 2000
```

```r
bootstrap.nonparam <- function(data, estimator, m=2000, ...) {
  bootstrap.samples <- lapply(1:m, function(x) sample(data, replace = TRUE))
  stats <- sapply(bootstrap.samples, function(y) estimator(y, ...))
  # Standard error
  se <- sd(stats)
  # 95 percentile confidence interval
  ci <- quantile(stats, probs = c(0.025, 0.975))
  list(se = se, ci = ci)
}

x.bs.median <-  bootstrap.nonparam(x, median)
x.bs.mean <-  bootstrap.nonparam(x, mean)
x.bs.meant <-  bootstrap.nonparam(x, mean, trim=0.05)

x.clean.bs.median <-  bootstrap.nonparam(x.clean, median)
x.clean.bs.mean <-  bootstrap.nonparam(x.clean, mean)
x.clean.bs.meant <-  bootstrap.nonparam(x.clean, mean, trim=0.05)

data.frame(
  Sample = c(rep("x", 3), rep("x.clean", 3)),
  Estimator = c("Median", "Mean", "Mean Trimmed") %>% rep(2),
  Standard.Error = c(x.bs.median$se, x.bs.mean$se, x.bs.meant$se,
                     x.clean.bs.median$se, x.clean.bs.mean$se,
                     x.clean.bs.meant$se) %>% sapply(round, 4),
  Confidence.Intervals=list(x.bs.median$ci, x.bs.mean$ci, x.bs.meant$ci,
                            x.clean.bs.median$ci, x.clean.bs.mean$ci,
                            x.clean.bs.meant$ci) %>%
    sapply(function(c) paste(c[1] %>% round(4), ", ", c[2] %>% round(4)))
) %>%
  print() %>%
  kable()
```

```
##    Sample    Estimator Standard.Error Confidence.Intervals
## 1       x       Median         0.0276    -0.0466 ,  0.0616
## 2       x         Mean         0.0258     0.0338 ,  0.1358
## 3       x Mean Trimmed         0.0233    -0.0079 ,  0.0823
## 4 x.clean       Median         0.0273     -0.067 ,  0.0384
## 5 x.clean         Mean         0.0224    -0.0491 ,  0.0396
## 6 x.clean Mean Trimmed         0.0228    -0.0488 ,  0.0421
```

Sample	Estimator	Standard.Error	Confidence.Intervals
x	Median	0.0276	-0.0466 , 0.0616
x	Mean	0.0258	0.0338 , 0.1358
x	Mean Trimmed	0.0233	-0.0079 , 0.0823
x.clean	Median	0.0273	-0.067 , 0.0384
x.clean	Mean	0.0224	-0.0491 , 0.0396
x.clean	Mean Trimmed	0.0228	-0.0488 , 0.0421

In my `bootstrap.nonparam()` function I use lapply to sample from the originally given data m times and apply the given estimator, i.e. `median`, `mean` or `mean` with an extra parameter. I then compute the standard error and the 95% quantiles.

Parametric bootstrap

```
# Function to perform parametric bootstrap and calculate statistics
parametric_bootstrap <- function(data, estimator, center, scale, m=2000, ...) {
  # generate samples
  bootstrap.samples <- lapply(1:m, function(x) rnorm(m, mean = center, sd = scale))
  # get stats (mean, median, ...)
  stats.estimated <- sapply(bootstrap.samples, function(y) estimator(y, ...))
  # get the stat of the original sample
  stat.original = estimator(data, ...)
  # get bias, standard error, and 95 percentile confidence interval
  bias <- mean(stats.estimated) - stat.original
  se <- sd(stats.estimated)
  ci <- quantile(stats.estimated, probs = c(0.025, 0.975))
  bias_corrected_estimate <- stat.original - bias

  list(bias = bias, se = se, ci = ci, bias_corrected_estimate = bias_corrected_estimate)

}


x.center <- mean(x)
x.scale <- sd(x)
```

```r
x.clean.center <- mean(x.clean)
x.clean.scale <- sd(x.clean)


x.bsp.median <-  parametric_bootstrap(x, median, x.center, x.scale)
x.bsp.mean <-  parametric_bootstrap(x, mean, x.center, x.scale)
x.bsp.meant <-  parametric_bootstrap(x, mean, x.center, x.scale, trim=0.05)

x.clean.bsp.median <-  parametric_bootstrap(x.clean, median,
                                            x.clean.center, x.clean.scale)
x.clean.bsp.mean <-  parametric_bootstrap(x.clean, mean, x.clean.center,
                                          x.clean.scale)
x.clean.bsp.meant <-  parametric_bootstrap(x.clean, mean, x.clean.center,
                                           x.clean.scale, trim=0.05)

data.frame(
  Sample = c(rep("x", 3), rep("x.clean", 3)),
  Estimator = c("Median", "Mean", "Mean Trimmed") %>% rep(2),
  Standard.Error = c(x.bs.median$se, x.bs.mean$se, x.bs.meant$se,
                     x.clean.bs.median$se, x.clean.bs.mean$se,
                     x.clean.bs.meant$se) %>% sapply(round, 4),
  Confidence.Intervals=list(x.bs.median$ci, x.bs.mean$ci, x.bs.meant$ci,
                            x.clean.bs.median$ci, x.clean.bs.mean$ci,
                            x.clean.bs.meant$ci) %>%
    sapply(function(c) paste(c[1] %>% round(4), ", ", c[2] %>% round(4))),
  Bias.Corrected.Estimate = c(x.bsp.median$bias_corrected_estimate,
                              x.bsp.mean$bias_corrected_estimate,
                              x.bsp.meant$bias_corrected_estimate,
                              x.clean.bsp.median$bias_corrected_estimate,
                              x.clean.bsp.mean$bias_corrected_estimate,
                              x.clean.bsp.meant$bias_corrected_estimate)
) %>%
  kable()
```

Sample	Estimator	Standard.Error	Confidence.Intervals	Bias.Corrected.Estimate
x	Median	0.0276	-0.0466 , 0.0616	-0.0618280
x	Mean	0.0258	0.0338 , 0.1358	0.0832604
x	Mean Trimmed	0.0233	-0.0079 , 0.0823	-0.0109918
x.clean	Median	0.0273	-0.067 , 0.0384	-0.0275045
x.clean	Mean	0.0224	-0.0491 , 0.0396	-0.0062052
x.clean	Mean Trimmed	0.0228	-0.0488 , 0.0421	0.0032899

As expected, the bias-corrected mean estimate of the medians is more different from its actual mean estimate, than is the case in the clean sample, and the standard error is also higher when having outliers. The bias corrected estimates differ noticeably from the simulations using the clean sample without outliers and the sample with outliers, for all estimated statistics. The standard errors of the estimates seemingly do not differ much adding outliers to the samples, but the confidence intervals appear to be slightly wider when usign outliers, for all statistics.

Task 3

Bootstrapping is a method where we repeatedly sample from our data to estimate the accuracy of a measurement, like the mean or median. By taking many samples with replacement, we can calculate a confidence interval for our estimate based on the variation across these samples. This is called non-parametric bootstrapping and works without assuming a specific data shape.

In parametric bootstrapping, we assume the data follows a certain distribution, like normal, and fit the data to get parameters (mean, standard deviation). We then generate samples based on this fitted model and calculate our statistic for each sample to build a confidence interval. Both types of bootstrapping give us insight into the stability and reliability of our estimates.